REMARKS

Claim Rejections – 35 USC § 103(a)

The Examiner continues to reject claims 1-12 as obvious over the combination of Lin and Nilsen. Applicant respectfully submits that claim 1, for at least the following reasons, traverses the Examiner's rejections.

In paragraph 6 of his latest Office Action, the Examiner states:

> It would have been obvious to one of ordinary skill in the art to combine Lin with Nilsen since the method of combining concurrent control flow graphs into sequential control flow graphs disclosed by Lin takes into account context switching between separate processes or threads, but fails to specify exactly how the code translation is generated, or what features are implemented therein.

The Examiner is factually incorrect when he states that "Lin takes into account context switching between separate processes or threads." Rather than addressing the problem of concurrent control flow graph (CCFG) to sequential control flow graph (SCFG) conversion with context switching, Lin duplicates portions (or subgraphs) of the concurrent control flow graph. Applicant shall refer to Lin's technique as "subgraph duplication." While one can attempt to reduce the amount of subgraph duplication in the resulting SCFG, by searching for a more optimal scheduling, there are many instances of CCFGs for which no schedule can avoid such duplication.

Applicant will now discuss an example CCFG, from the application, where Lin's approach results in subgraph duplication while applicant's approach, of using context switching, does not.

Applicant refers the Examiner to the example CCFG of Figure 7. The scheduling of each node of Figure 7 is indicated by the numbers, next to each node, enclosed in parentheses. The nodes of Figure 7 will be referred to by their scheduling numbers.

Figure 7 can be described as being comprised of three subgraphs. The first subgraph is comprised of nodes 2 and 3. The second subgraph is comprised of nodes 4 and 5. The third subgraph is comprised of nodes 6 and 7. The first and third subgraphs are part of a first thread while the second subgraph is part of a second thread. According to the scheduling indicated in Figure 7, the first subgraph is executed first, the second subgraph is executed second and the third subgraph is executed third.

Figure 8Q shows an SCFG produced from Figure 7 in accordance with the teachings of the applicant's invention. Figure 8Q operates as follows, where a variable

"th_1009.stateVar" is dedicated to containing the state of the first thread while a variable "th_1010.stateVar" is dedicated to containing the state of the second thread.

After the first subgraph is executed, Figure 8Q shows the state of the first thread being saved in state variable th_1009.stateVar. Specifically, after conditional "A" is executed, control can pass to either branch "CA1" or "CA2." If branch CA1 is taken, a value of "8" is saved in th_1009.stateVar. If branch CA2 is taken, "emit" statement "B" is executed, after which the value "6" is saved in th_1009.stateVar.

The second subgraph is executed with the state of its second thread indicated by state variable "th_1010.stateVar."

After the second subgraph is executed, the first thread is restarted by reading the value stored in th_1009.stateVar. The reading of th_1009.stateVar is accomplished by node "rn_1047" (where "rn_1047" is a type of node referred to in the application as a "restart node"). If the value stored in th_1009.stateVar is "6," the first thread is continued by executing conditional "C" (also indicated in Figure 8Q as node 1045). If the value stored in th_1009.stateVar is not "6," the first thread is continued by executing node 1058.

In contrast, attached to the present response is a Figure 11 that depicts an SCFG resulting from application of Lin's approach to Figure 7. Because Lin uses subgraph duplication, rather than context switching, the second subgraph (labeled as nodes 4 and 5) is duplicated.

Since there is no alternative scheduling for the nodes of Figure 7, other than the sequence indicated in Figure 7, there is no scheduling for the nodes of Figure 7 that can eliminate the subgraph duplication of Figure 11.

Having clarified, by means of an example, how the applicant's claimed invention (that uses context switches with thread-dedicated state variables) differs from Lin's approach of using subgraph duplication, applicant respectfully submits the rejections of claim 1 are traversed.

Since claims 2-9 are dependent on claim 1, claims 2-9 are allowable for at least the same reasons. Since claims 10-12 are similar to claim 1, with the exception of their data processing system, computer program product and computer data signal form, such claims are also allowable for at least the same reasons.

Summary

Applicant respectfully submits that all 35 USC § 103 rejections have been traversed. Therefore, applicant requests a Notice of Allowance be granted.

The U.S. Patent and Trademark Office is hereby authorized to charge any fee deficiency, or credit any overpayment, to Deposit Account No. 502584 referencing docket number 06816.0158.

Respectfully submitted,

Jonathan T. Kaplan
Registration No. 38,935

Date: Feb. 4, 2004

JONATHAN T. KAPLAN
ATTORNEY AT LAW
140 NASSAU STREET
NEW YORK, NY 10038-1501
TEL. 917-674-5017